Textual Analysis

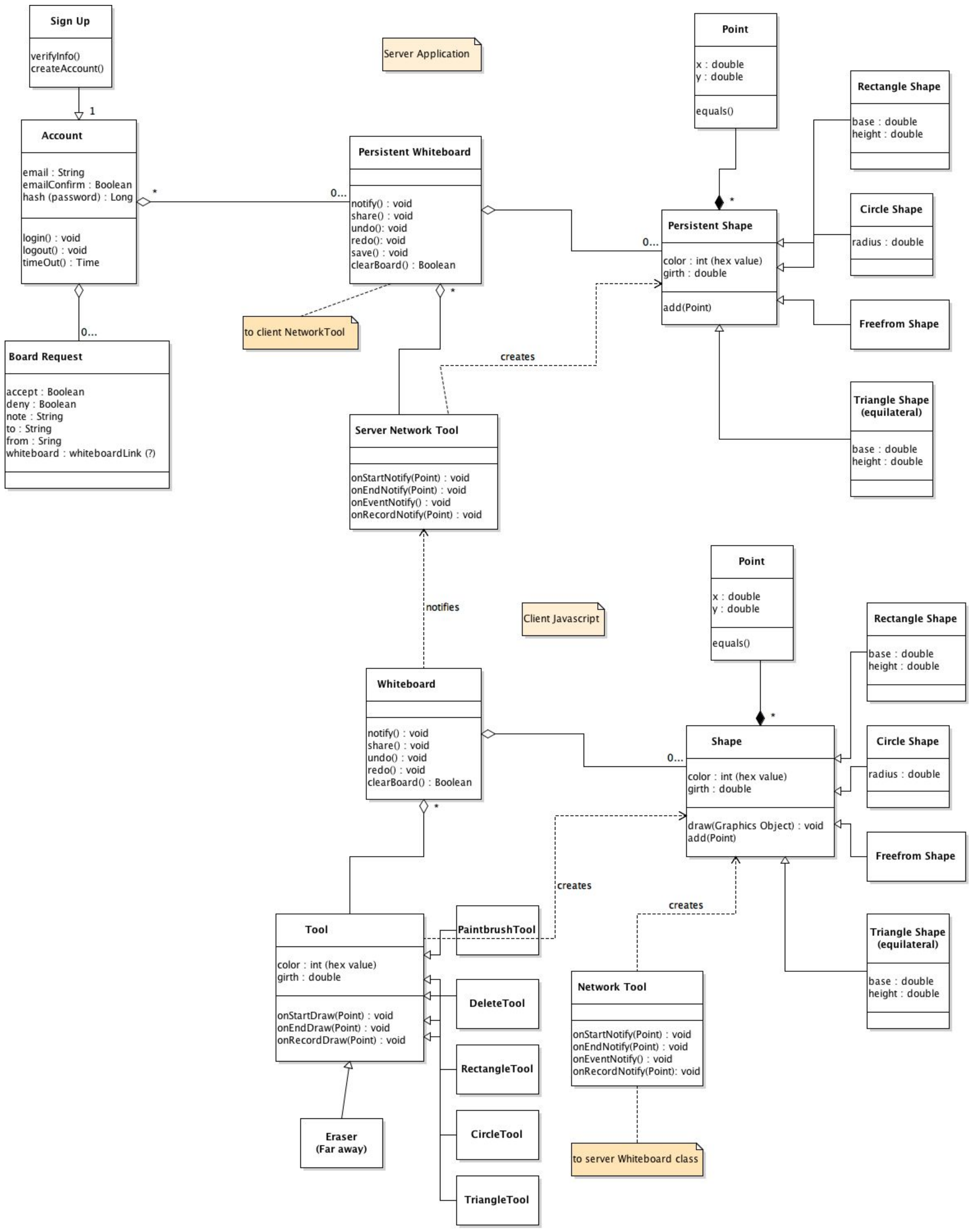
CS320 - 103: Software Engineering, Spring Semester 2016

Due Date: 2-22-2016

Cara Sperbeck, Alexander Smith, Aaron Walsh

Methods Classes Attributes Unused Interfaces

Nouns	Verbs
● Account	● Accept
● Account access	● Add
● Board Request	● Clear
● Color	● Confirm
● Delete	● Create (account)
● Email	● Delete
● Freeform	● Deny
● Girth	● Draw
● Google re-captcha	● Erase
● Hash(password)	● Invite (to edit)
● Image	● Load
● Invitation	● Login
● Paintbrush	● Logout
● Point	● Notify
● Public	● Redo
● Shape	● Save (account info/whiteboards)
● Tools (shape/select/line/color select/eraser)	● Select
● Whiteboard	● Sign up
	● Share
	● Undo
	● Verify


Sign Up
verifyInfo()
createAccount()
1
Account
email : String
emailConfirm : Boolean
hash (password) : Long
login() : void
logout() : void
timeOut() : Time
*
0...
Board Request
accept : Boolean
deny : Boolean
note : String
to : String
from : Sring
whiteboard : whiteboardLink (?)
Server Application
Persistent Whiteboard
notify() : void
share() : void
undo(): void
redo(): void
save() : void
clearBoard() : Boolean
0...
*
to client NetworkTool
creates
Server Network Tool
onStartNotify(Point) : void
onEndNotify(Point) : void
onEventNotify() : void
onRecordNotify(Point) : void
Point
x : double
y : double
equals()
*
Persistent Shape
color : int (hex value)
girth : double
add(Point)
0...
Rectangle Shape
base : double
height : double
Circle Shape
radius : double
Freefrom Shape
Triangle Shape (equilateral)
base : double
height : double
notifies
Client Javascript
Whiteboard
notify() : void
share() : void
undo() : void
redo() : void
clearBoard() : Boolean
*
Point
x : double
y : double
equals()
*
Shape
color : int (hex value)
girth : double
draw(Graphics Object) : void
add(Point)
0...
Rectangle Shape
base : double
height : double
Circle Shape
radius : double
Freefrom Shape
Triangle Shape (equilateral)
base : double
height : double
creates
creates
Tool
color : int (hex value)
girth : double
onStartDraw(Point) : void
onEndDraw(Point) : void
onRecordDraw(Point) : void
Eraser (Far away)
PaintbrushTool
DeleteTool
RectangleTool
CircleTool
TriangleTool
Network Tool
onStartNotify(Point) : void
onEndNotify(Point) : void
onEventNotify() : void
onRecordNotify(Point): void
to server Whiteboard class

Our UML model is split into two major sections: Server and Client. These segments are used to distinguish between parts of the application that will be implemented using differing languages and technology. A user interacts directly with the Client side of the model, which will be implemented in Javascript and will be running in the user's web browser. The server will be written in some form of Java and run on the Server computer. The server stores shapes, holds account information, and talks to other connected clients. When Clients draw, they use tools, which create shape objects, which are drawn and appear on their whiteboard. The Whiteboard class contains the drawn shapes and user accounts that are connected and has the behavior to talk between client and server.

In our Server section of the UML model, the core class is ’Persistent Whiteboard’. This ‘Persistent Whiteboard’ object contains references to nearly all the other classes created in our UML analysis. Unlike in the Client model, the data stored here persists even when the user reloads the page or leaves. The server portion also contains objects for storing account information and has behaviors related to accounts and their creation, such as login(), logout(), createAccount(), etc.

In the client section of our UML model, the core class is ‘Whiteboard’ which handles immediate drawing events from the user and sends them to the “Server Network Tool” class for the events to be stored in the server. The whiteboard class contains a list of references to shape objects and tool objects. These objects are sent in the notify() method to the server to be saved and will persist. There are several methods related to the drawing canvas, such as undo(), redo() and clearBoard().

The generic shape class in our model takes points and stores them in a list. It also has a generic method for drawing. This method will be specific to the type of shape being drawn. These are collections of points with some behaviors as to how they should be arranged and drawn. We decided to differentiate this from tool, which handles shape creation because tools do not need to hold reference to the shapes they create. Instead, the central whiteboard or persistent whiteboard should hold references to the individual shapes.

Whenever a user draws on a whiteboard they draw using Tools, which record incoming mouse input and format it. This data is then stored as their respective shape objects. ‘Tool’ has fields related to how the respective shape is to be drawn in graphics, such as girth (representing line weight or thickness) and color. The onStartDraw(Point) method is called to store the first point of a shape and begin a drawing object. Similarly, the onEndDraw(Point) records the last point of the drawing and closed the shape object. While the user is drawing, the onRecordDraw(Point) is recording each point as they move the tool around the Whiteboard canvas. The subclasses of tool must override these generic methods and insert their own specific behavior related to the type of shape they are to be creating. The tool object and its respective subclasses are not present in the server model.

Network tool is a specific type of tool that formats incoming network messages and packages them into shapes, just like the client side drawing tools. There is one of these interfaces in each section of the model. It has methods like the on...Draw(Point) in tool, but instead they are called when a network package comes from either the server or the client and must be packaged into a shape, like how tool packages shapes.